Foundations of Cryptography. Lecture 1: Minicrypt

Anna Lysyanskaya

Modern Cryptography: Note on Methodology

- A recipe for a secure system:
  - (1) Define how it should work for the honest participants
  - (2) Define what you mean by “secure”
  - (3) Give an algorithm and prove that it satisfies the definition of security
    - Usually, this proof requires some computational assumptions, such as $P \neq NP$

- Is this obvious?
  - Yes, in hindsight!
  - Earliest cryptography research (prior to 1980) didn’t do this
  - First examples: Goldwasser-Micali 1982 definition of security for encryption
    - They won the Turing award for this in 2012

- Why is this important?
  - Can’t just say “look, it works!” You have to prove that it can’t be broken.

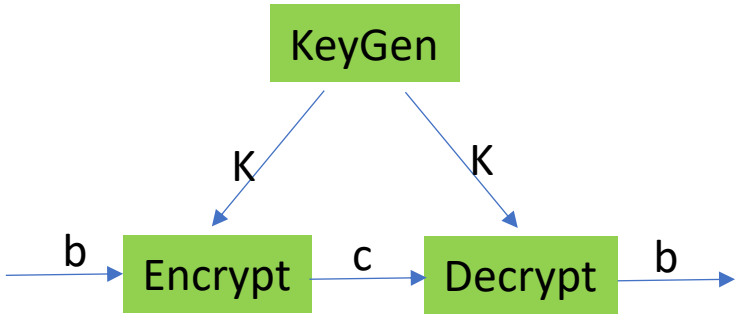
Сучасна криптографія: методологія

- Формула безпечної системи:
  - (1) Визначте, як це має працювати для законних учасників
  - (2) Визначте, що ви маєте на увазі під «безпечним»
  - (3) Опишіть алгоритм і доведіть, що він задовольняє визначення безпеки
    - Зазвичай для цього доказу потрібні деякі обчислювальні припущення, наприклад $P \neq NP$

- Можливо, це й очевидно?
  - Так, оглядаючись назад!
  - Найдавніші дослідження криптографії (до 1980 року) цього не робили
  - Перші приклади: Голдвассер-Мікалі 1982 визначення безпеки для шифрування
    - У 2012 році вони отримали за це премію Тюрінга

- Чому це важливо?
  - Не можна просто сказати: «Подивіться, це працює!» Ви повинні довести, що вашу систему неможливо зламати.

Example: secure encryption

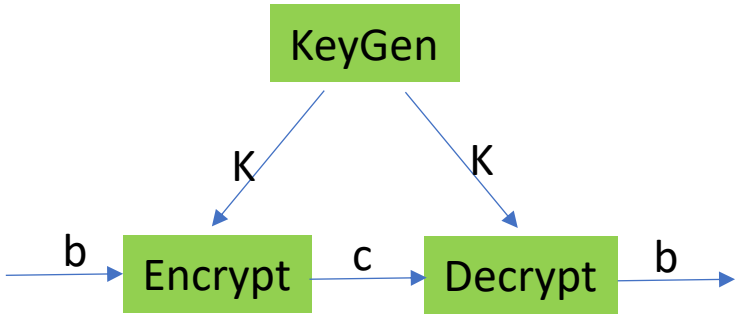


- (Symmetric-key) encryption system:
  - KeyGen algorithm selects a key K
  - Encrypt(key K, bit b) -> ciphertext c
  - Decrypt(key K, ciphertext c) -> bit b

- Honest participants need to recover the message, i.e.: ∀ b ∈ {0,1} and ∀ K, if ciphertext c was output by Encrypt(K,b), then Decrypt(K,c) outputs b

- This was step 1 in our recipe: we have defined how the system must work for the honest participants! Now the hard parts: steps 2 & 3.

Example: secure encryption



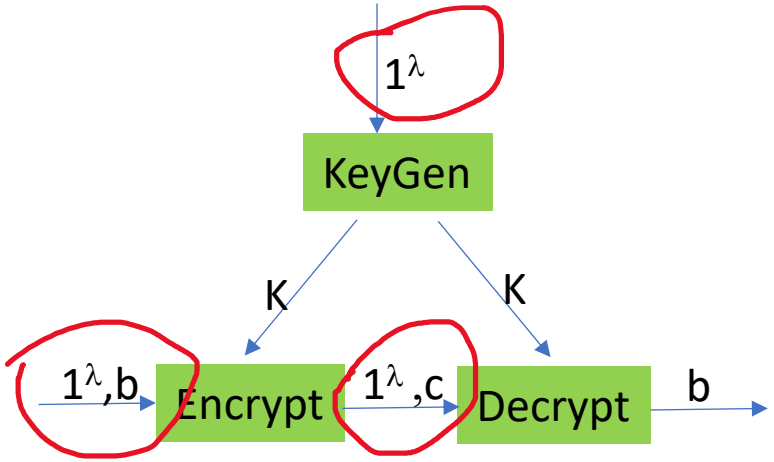
- (Symmetric-key) encryption system:
  - KeyGen selects a key K
  - Encrypt(key K, bit b) -> ciphertext c
  - Decrypt(key K, ciphertext c)-> bit b

- Define what we mean by secure: Визначте, що є «безпечність»
  - Intuition: to the adversary, Encrypt(K,0) looks the same as Encrypt(K,1)
    Для противника, Encrypt(K,0) виглядає так само, як Encrypt(K,1)
  - Who is the adversary? Хто наш противник?
    - What are the adversary's computational resources? Які в нього обчислювальні ресурси?
    - What are its input and outputs? Які в нього вхідні та вихідні дані?
    - How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - What does it mean to "look the same"? Що значить виглядати так само/однаково?

Who Is the Adversary? Хто наш противник?

Who Is the Adversary?

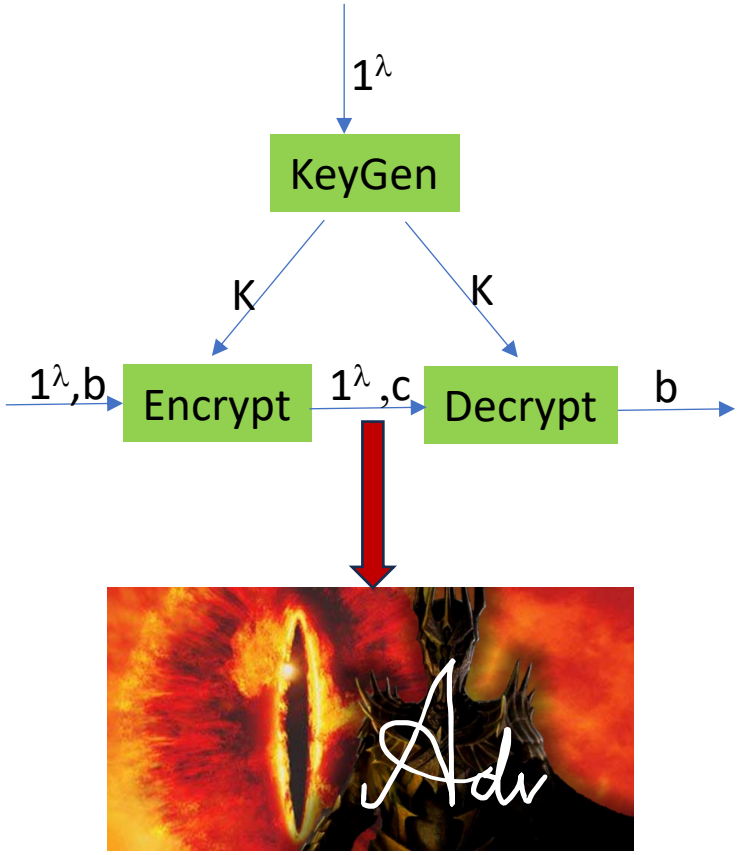
- What are the adversary's computational resources?
  Які в нього обчислювальні ресурси?
  - Think worst-case! Давайте думати про найгірший випадок!
  - Traditional complexity-theoretic approach: we have a security parameter λ. The honest participants run in time that's fast in λ. The adversary can take any polynomial time in λ. The adversary can be a probabilistic algorithm.



  - Other approaches: concrete security, e.g. think of the adversary as a circuit of a large specific size.

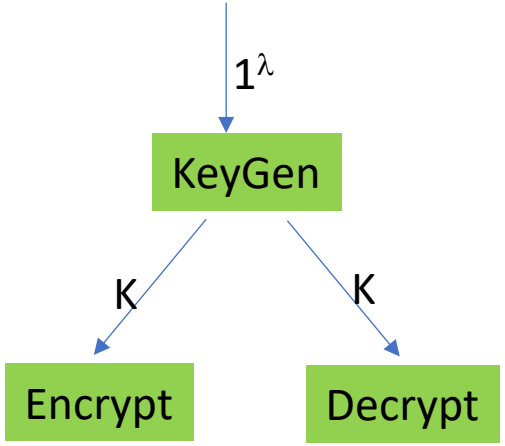
Who Is the Adversary?

- What are its input and outputs? Які в нього вхідні та вихідні дані?
  - Think worst-case! E.g., adversary knows everything except key K and bit b.



Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.



Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.

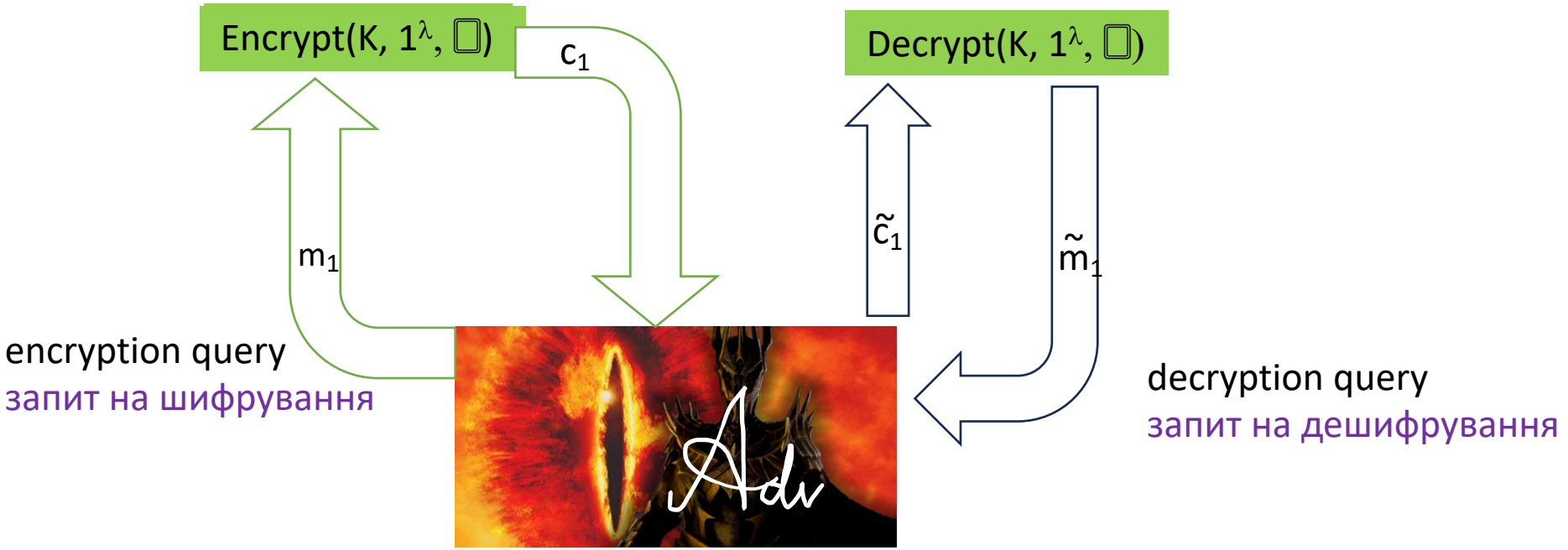
black boxes/oracles for encryption and decryption
чорні скриньки/оракули
для шифрування та дешифрування

Encrypt(K, 1^λ, □)

Decrypt(K, 1^λ, □)

Who Is the Adversary?

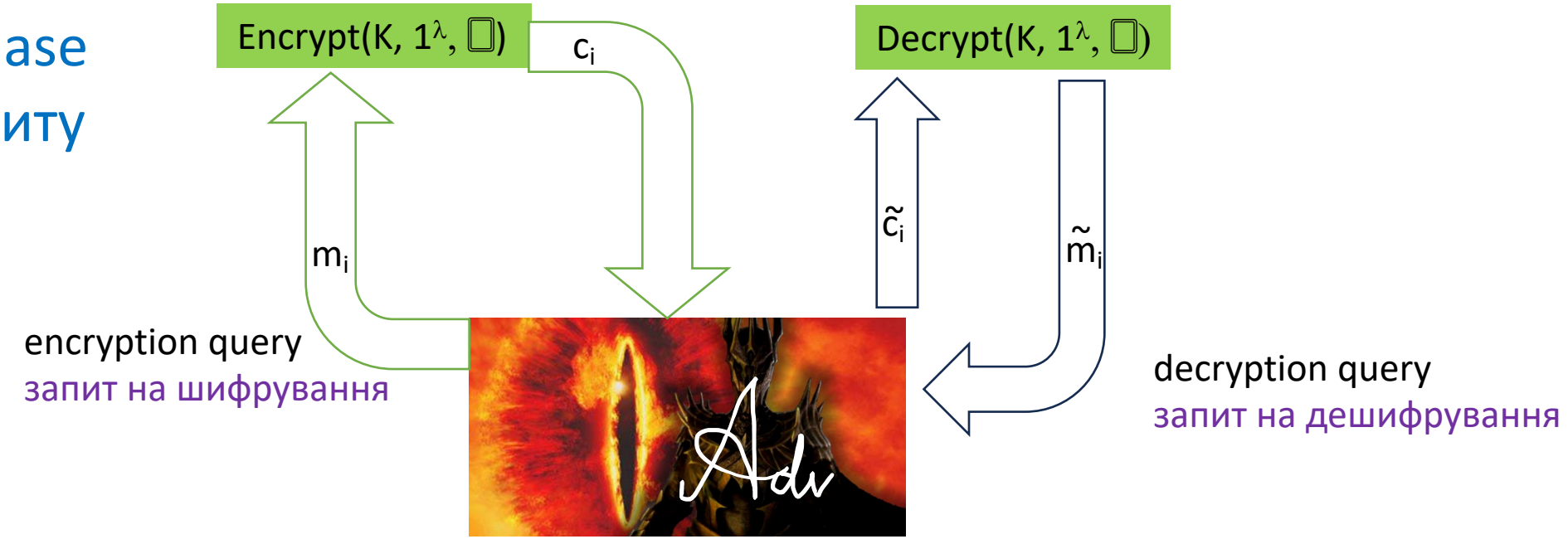
- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.



Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.

Query phase
Фаза запиту

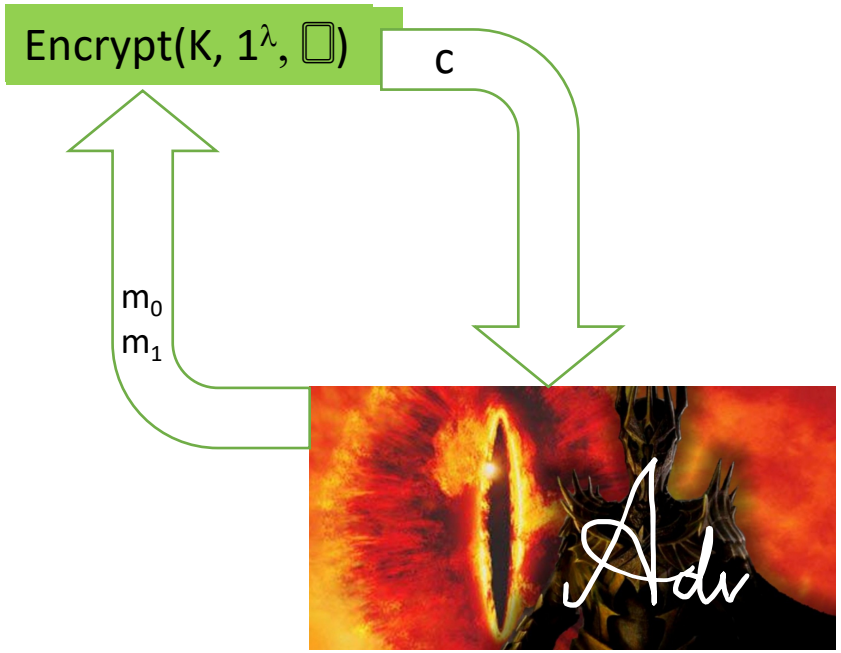


Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.

Challenge phase

Фаза виклику



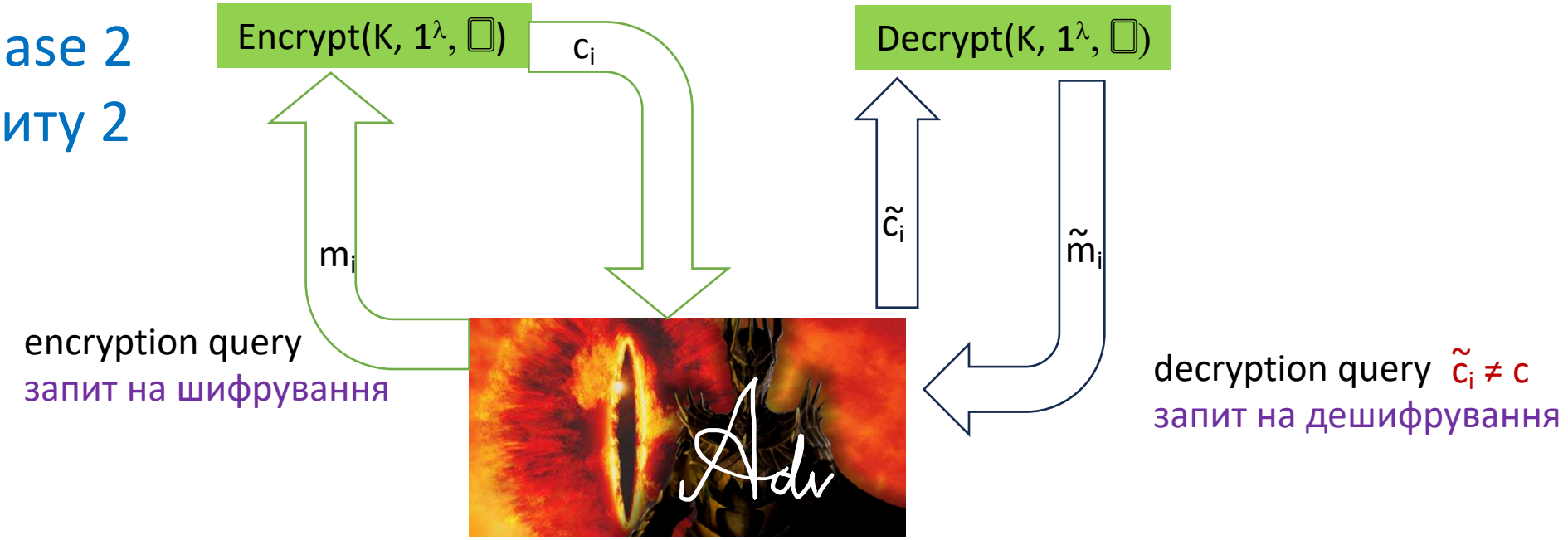
Length of m is denoted |m|
We require that $|m_0|=|m_1|$

Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.

Query phase 2

Фаза запиту 2



Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.

Output phase

Вихідна фаза

Who Is the Adversary?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.
  - Summary: in the worst-case scenario
    - The system is set up with a key K chosen using KeyGen(1^λ)
    - Adversary A(1^λ) runs in polynomial time in λ and can query Encrypt(1^λ,K,.) and Decrypt(1^λ, K,.) to its heart's content
    - Adversary produces two challenge messages, m_0 and m_1, of the same length
    - Adversary receives a challenge c <- Encrypt(1^λ,K,m_b)
    - Then A can query Encrypt and Decrypt some more
    - Finally adversary produces an output

Who Is the Adversary? What Is Security?

- How does the adversary interact with other system participants? Як противник взаємодіє з іншими учасниками системи?
  - Think worst-case! The adversary observes the system long-term, controls many of the inputs. Противник довго спостерігає за системою, контролює багато вхідних даних.
  - Summary: in the worst-case scenario parameterized by bit b
    - The system is set up with a key K chosen using KeyGen(1^λ)
    - Adversary A(1^λ) runs in polynomial time in λ and can query Encrypt(1^λ,K,.) and Decrypt(1^λ, K,.) to its heart's content
    - Adversary produces two challenge messages, m_0 and m_1, of the same length
    - Adversary receives a challenge c <- Encrypt(1^λ, K,m_b)
    - Then A can query Encrypt and Decrypt some more
    - Finally adversary produces an output
- What does it mean to "look the same"? Що значить ви само/однаково?
  - Adversary's output is the same in case b=0 as in the case b=1 Вихід противника у випадку b=0 такий самий, як і у вип. b=1

What Does Security Mean? Що є безпекою?

- What does it mean to “look the same”? Що значить виглядати так само/однаково?
  - Adversary’s output is the same in case b=0 as in the case b=1 / Вихід противника такий самий у випадку b=0 і у випадку b=1
  - That’s too strong a requirement: what if A always just outputs the ciphertext c? Це надто сувора вимога: що, якщо A завжди просто виводить зашифрований текст c?

What Does Security Mean? Що є безпекою?

- What does it mean to “look the same”? Що значить виглядати так само/однаково?
  - A better approach: measure the probability that A outputs, say, 0.
    Кращий підхід: виміряти ймовірність того, що A виведе, скажімо, 0.

p_0 = Pr[A outputs 0 when b=0] = Pr[A виведе 0 коли b=0]
p_1 = Pr[A outputs 0 when b=1] = Pr[A виведе 0 коли b=1]

Encryption scheme is secure if $p_0=p_1$

What Does Security Mean? Що є безпекою?

- What does it mean to “look the same”? Що значить виглядати так само/однаково?

  - A better approach: measure the probability that A outputs, say, 0.
    Кращий підхід: виміряти ймовірність того, що A виведе, скажімо, 0.

p_0 = Pr[A outputs 0 when b=0] = Pr[A виведе 0 коли b=0]
p_1 = Pr[A outputs 0 when b=1] = Pr[A виведе 0 коли b=1]

Encryption scheme is secure if $p_0=p_1$

  - Do they have to be equal? That’s too strong a requirement: with some small probability, A can guess K and then will be able to distinguish…
    Чи мають вони бути рівними? Це надто сувора вимога: з деякою невеликою ймовірністю A може вгадати K, а потім зможе розрізнити…

What Does Security Mean? Що є безпекою?

- What does it mean to “look the same”?
  Що значить виглядати так само/однаково?

p_0 = Pr[A outputs 0 when b=0] = Pr[A виведе 0 коли b=0]
p_1 = Pr[A outputs 0 when b=1] = Pr[A виведе 0 коли b=1]

Encryption scheme is secure if $|p_0 - p_1| = \text{negligible}(\lambda)$

- Negligible function: $\nu(\lambda)$ is a negligible function if for all c there exists λ_c such that for all $\lambda > \lambda_c$, $\nu(\lambda) < \lambda^{-c}$. Examples: $2^{-\lambda}$, $\lambda^{-\log \lambda}$. Незначна функція
  - Intuition: if in an experiment, an event’s probability is negligible, and you only carry out a polynomial number of (independent) experiments, you are extremely unlikely to observe the event happening
    Інтуїтивно: якщо в експерименті ймовірність події є незначною, і ви виконуєте лише поліноміальну кількість (незалежних) експериментів, ви навряд чи спостерігатимете цю подію

What Does Security Mean? Що є безпекою?

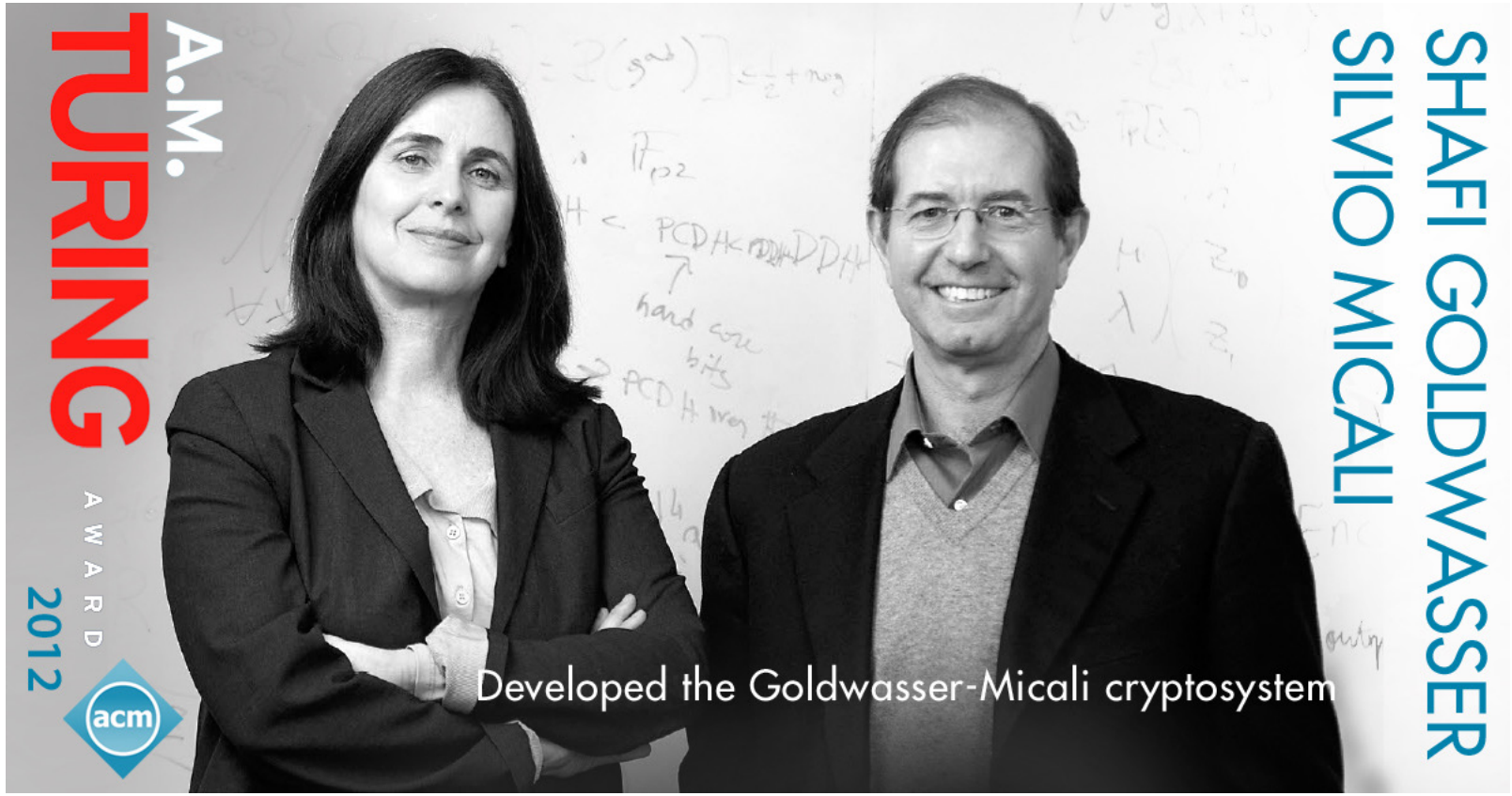
- What does it mean to “look the same”?
  Що значить виглядати так само/однаково?

p_0 = Pr[A outputs 0 when b=0] = Pr[A виведе 0 коли b=0]
p_1 = Pr[A outputs 0 when b=0] = Pr[A виведе 0 коли b=1]

Encryption scheme is secure if $|p_0 - p_1| = \text{negligible}(\lambda)$

- This is called “indistinguishability,” and Goldwasser and Micali won the Turing award for formulating this definition back in the 1980s
  Це називається «нерозрізнення», і Голдвассер і Мікалі отримали премію Тюрінга за формулювання цього визначення в 1980х роках

What Does Security Mean? Що є безпекою?



What Does Security Mean? Що є безпекою?

- What does it mean to “look the same”?
  Що значить виглядати так само/однаково?

p_0 = Pr[A outputs 0 when b=0] = Pr[A виведе 0 коли b=0]
p_1 = Pr[A outputs 0 when b=1] = Pr[A виведе 0 коли b=1]

Encryption scheme is secure if $|p_0 - p_1| = \text{negligible}(\lambda)$

- This is called “indistinguishability,” and Goldwasser and Micali won the Turing award for formulating this definition back in the 1980s
  Це називається «нерозрізнення», і Голдвассер і Мікалі отримали премію Тюрінга за формулювання цього визначення в 1980х роках

Back to Our Recipe for Secure Encryption
Назад до нашої формули створення безпечного шифрування

- A recipe for a secure system:
  - (1) Define how it should work for the honest participants -- DONE
  - (2) Define what you mean by “secure” -- DONE
  - (3) Give an algorithm and prove that it satisfies the definition of security – TODO

- Формула безпечної системи:
  - (1) Визначте, як це має працювати для законних учасників -- зроблено
  - (2) Визначте, що ви маєте на увазі під «безпечним» -- зроблено
  - (3) Опишіть алгоритм і доведіть, що він задовольняє визначення безпеки -- потрібно зробити

- Usually, this proof requires some computational assumptions, such as P ≠ NP

  Зазвичай для цього доказу потрібні деякі обчислювальні припущення, наприклад P ≠ NP

Why Assumptions Are Necessary / Чому потрібні припущення?

- Suppose P = NP
- Then after some number of queries, the adversary will find the key K / Тоді після певної кількості запитів противник знайде ключ K
  - How? Homework exercise!
  - Hint: the adversary asked encryption and decryption queries, and received answers. He is looking for a key that is consistent with these queries and their answers. Can you think of a series of yes/no questions about the key K whose answers will enable him to find it? If P=NP, then all these questions can be efficiently answered (why?) Підказка: противник ставив запити на шифрування та дешифрування та отримував відповіді. Він шукає ключ, який узгоджується з цими запитами та відповідями на них. Чи можете ви придумати серію запитань «так/ні» про ключ K, відповіді на які дозволять йому знайти його? Якщо P=NP, то на всі ці запитання можна ефективно відповісти (чому?)
- As soon as A finds the key K, he can distinguish an encryption of m_0 from that of m_1.

What Assumption Is Necessary and Sufficient?
Яке припущення є необхідним і достатнім?

Theorem: Secure encryption exists if and only if one-way functions exist.

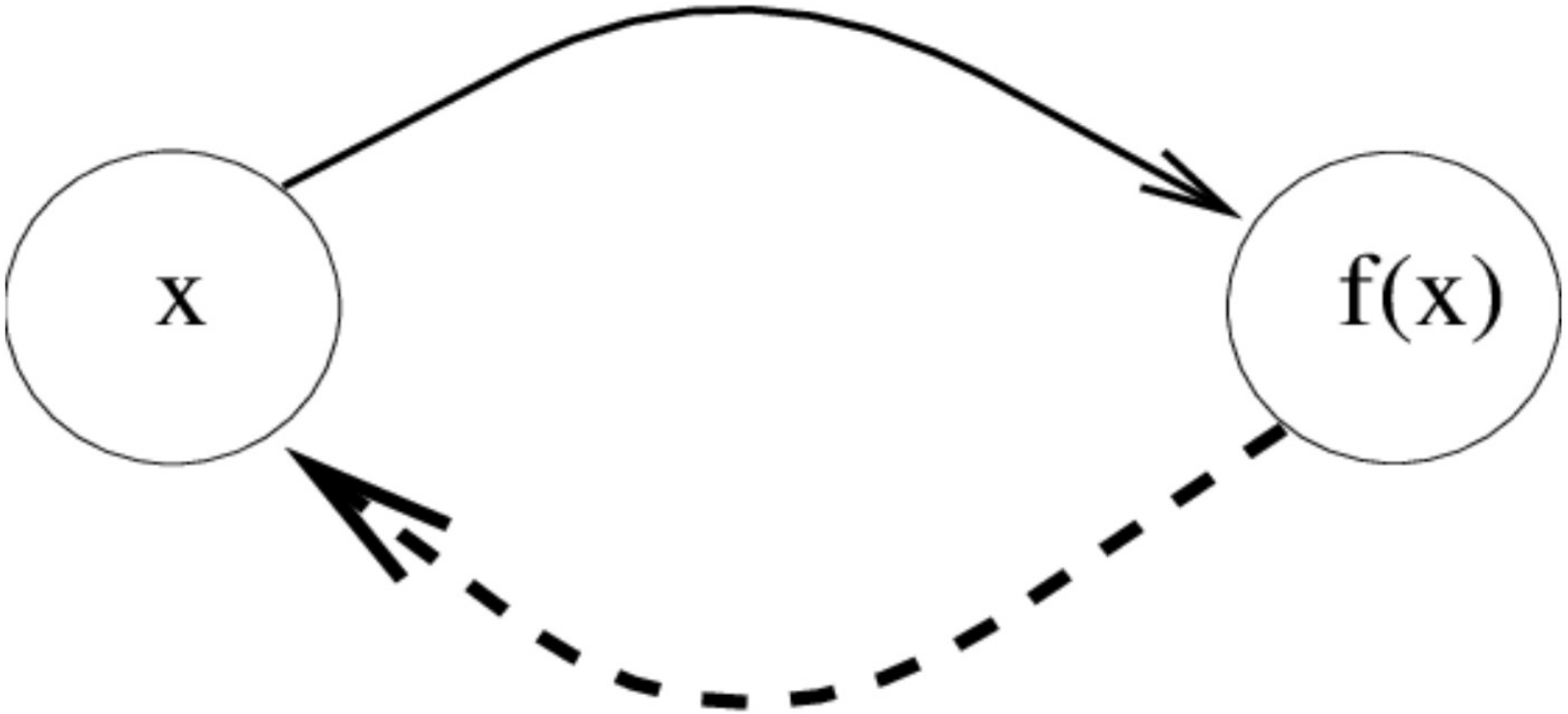
Теорема: Безпечне шифрування існує якщо і тільки якщо існують односторонні функції.

What is a one-way function? Що таке одностороння функція?

One-Way Function (OWF) / Одностороння функція

- Definition: Let $f : \{0,1\}^k \to \{0,1\}^{n(k)}$ be a polynomial-time computable function defined for every k. For an adversary A and an integer k, consider the following experiment: x is picked uniformly at random from $\{0,1\}^k$, then A runs on input f(x), and outputs x'. Let $E_{A,k}$ be the event that f(x') = f(x).
  f is a one-way function if for all probabilistic polynomial-time adversaries A, $Pr[E_{A,k}]$ is negligible in k
  Визначення: Нехай $f : \{0,1\}^k \to \{0,1\}^{n(k)}$ — функція, обчислювана за поліноміальним часом, визначена для кожного k. Для противника A та цілого числа k розглянемо наступний експеримент: x вибирається рівномірно випадковим чином із $\{0,1\}^k$, потім A вводить f(x) і виводить x'. Нехай $E_{A,k}$ — випадок, що f(x') = f(x).
  f є односторонньою функцією, якщо для всіх поліноміальних супротивників A, ймовірність $Pr[E_{A,k}]$ є незначною у k

One-Way Function (OWF) / Одностороння функція



One-Way Function (OWF) / Одностороння функція

- Definition: Let $f : \{0,1\}^k \to \{0,1\}^{n(k)}$ be a polynomial-time computable function defined for every k. For an adversary A and an integer k, consider the following experiment: x is picked uniformly at random from $\{0,1\}^k$, then A runs on input f(x), and outputs x'. Let $E_{A,k}$ be the event that f(x') = f(x).
  f is a one-way function if for all probabilistic polynomial-time adversaries A, $Pr[E_{A,k}]$ is negligible in k
  Визначення: Нехай $f : \{0,1\}^k \to \{0,1\}^{n(k)}$ — функція, обчислювана за поліноміальним часом, визначена для кожного k. Для противника A та цілого числа k розглянемо наступний експеримент: x вибирається рівномірно випадковим чином із $\{0,1\}^k$, потім A вводить f(x) і виводить x'. Нехай $E_{A,k}$ — випадок, що f(x') = f(x).
  f є односторонньою функцією, якщо для всіх поліноміальних супротивників A, ймовірність $Pr[E_{A,k}]$ є незначною у k

One-Way Function (OWF) / Одностороння функція

- Definition: Let $f : \{0,1\}^k \to \{0,1\}^{n(k)}$ be a polynomial-time computable function defined for every k. For an adversary A and an integer k, consider the following experiment: x is picked uniformly at random from $\{0,1\}^k$, then A runs on input f(x), and outputs x'. Let $E_{A,k}$ be the event that f(x') = f(x).
  f is a one-way function if for all probabilistic polynomial-time adversaries A, $Pr[E_{A,k}]$ is negligible in k

- More compact notation for the same thing:
  f is a one-way function if for ppt A,

$$Pr[x \leftarrow \{0,1\}^k; x' \leftarrow A(f(x)) : f(x')=f(x)] = negl(k)$$

Experiment

Event $E_{A,k}$

One-Way Function (OWF) / Одностороння функція

- More compact notation for the same thing:
  Більш компактне позначення для того самого:
  f is a one-way function if for ppt A,

$$\Pr[x \leftarrow \{0,1\}^k;\ x' \leftarrow A(f(x)) : f(x')=f(x)] = \mathrm{negl}(k)$$

Experiment
Експеримент

Event $E_{A,k}$
Випадок $E_{A,k}$

One-Way Function (OWF) / Одностороння функція

- Definition: Let $f : \{0,1\}^k \to \{0,1\}^{n(k)}$ be a polynomial-time computable function defined for every k. For an adversary A and an integer k, consider the following experiment: x is picked uniformly at random from $\{0,1\}^k$, then A runs on input f(x), and outputs x'. Let $E_{A,k}$ be the event that f(x') = f(x).
  f is a one-way function if for all probabilistic polynomial-time adversaries A, $Pr[E_{A,k}]$ is negligible in k

- More compact notation for the same thing:
  f is a one-way function if for ppt A,

$$Pr[x \leftarrow \{0,1\}^k; x' \leftarrow A(f(x)) : f(x')=f(x)] = negl(k)$$

Experiment

Event $E_{A,k}$

Why Is It Necessary to Assume OWFs?
Чому необхідно припускати існування OWF?

- Secure encryption implies one-way functions.
  - The algorithm Encrypt(K,1^λ,m; random bits R) is a one-way function of K,R
  - Why? Homework exercise!

- Безпечне шифрування передбачає односторонні функції.
  - Алгоритм Encrypt(K,1^λ,m; random bits) є односторонньою функцією K,m,R
  - Чому? Домашня робота.

Why Is It Sufficient to Assume OWFs?
Чому достатньо припускати існування OWF?

- Can construct (symmetric) secure encryption from one-way functions.
  Ми знаємо, як побудувати (симетричне) безпечне шифрування з односторонніх функцій
- Roadmap / план:
  - Construct a pseudorandom generator from a one-way function
    будуємо псевдовипадковий генератор із OWF
  - Construct a pseudorandom function from a pseudorandom generator
    будуємо псевдовипадкову функцію з псевдовипадкового генератору
  - Construct a block cipher from a pseudorandom function
    будуємо блоковий шифр із псевдовипадкової функції
  - Construct secure symmetric encryption from a block cipher
    будуємо безпечне симетричне шифрування з блокового шифру

Warning: This is not how it works in practice
Попередження: у реальному житті це не так

- In practice, the cryptography research community spent decades on directly constructing a block cipher / На практиці протягом багатьох десятиліть співтовариство дослідників криптографії розробляло блоковий шифр, не проходячи цих етапів
  - Culminated in the design of AES, the advanced encryption standard
    Кульмінацією стала розробка AES, передового стандарту шифрування
  - Standardized by the National Institute of Standards (NIST), USA
    Стандартизовано Національним інститутом стандартів (NIST), США
  - (I just visited NIST and reviewed their work as part of a National Academy of Sciences study. Let me tell you, these people are AMAZING.)
    (Я щойно відвідала NIST і переглянула їхню роботу в рамках дослідження Національної академії наук. Дозвольте мені сказати вам, що ця організація ЧУДОВА.)
    https://www.nationalacademies.org/our-work/assessment-of-the-national-institute-of-standards-and-technology-nist-information-technology-laboratory-itl

Why Is It Sufficient to Assume OWFs?
Чому достатньо припускати існування OWF?

- Can construct (symmetric) secure encryption from one-way functions.
  Ми знаємо, як побудувати (симетричне) безпечне шифрування з односторонніх функцій
- Roadmap / план:
  - Construct a pseudorandom generator from a one-way function
    будуємо псевдовипадковий генератор із OWF
  - Construct a pseudorandom function from a pseudorandom generator
    будуємо псевдовипадкову функцію з псевдовипадкового генератору
  - Construct a block cipher from a pseudorandom function
    будуємо блоковий шифр із псевдовипадкової функції
  - Construct secure symmetric encryption from a block cipher
    будуємо безпечне симетричне шифрування з блокового шифру

Why Is It Sufficient to Assume OWFs?
Чому достатньо припускати існування OWF?

- Can construct (symmetric) secure encryption from one-way functions.
  Ми знаємо, як побудувати (симетричне) безпечне шифрування з односторонніх функцій
- Roadmap / план:
  - Construct a pseudorandom generator from a one-way function
    будуємо псевдовипадковий генератор із OWF
  - Construct a pseudorandom function from a pseudorandom generator
    будуємо псевдовипадкову функцію з псевдовипадкового генератору
  - Construct a block cipher from a pseudorandom function
    будуємо блоковий шифр із псевдовипадкової функції
  - Construct secure symmetric encryption from a block cipher
    будуємо безпечне симетричне шифрування з блокового шифру

Pseudorandom Generator: Definition
Псевдовипадковий генератор: визначення

- Idea: An algorithm G takes as input a (short) k-bit seed s and outputs a (long) 2k-bit string R that looks random
- What does “look random” mean? How do we define it?

Pseudorandom Generator: Definition
Псевдовипадковий генератор: визначення

- Idea: An algorithm G takes as input a (short) k-bit seed s and outputs a (long) 2k-bit string R that looks random
- What does “look random” mean? How do we define it?
- INDISTINGUISHABILITY!

Pseudorandom Generator: Definition
Псевдовипадковий генератор: визначення

- Let $G : \{0,1\}^k \rightarrow \{0,1\}^{2k}$ be an efficient algorithm that is defined for all k. G is a pseudorandom generator if for all ppt A, $|p_{G,A}(k)-p_{U,A}(k)|$ is negligible, where

$$p_{G,A}(k) = Pr[s \leftarrow \{0,1\}^k; R = G(s); b \leftarrow A(R) : b = 0]$$
$$p_{U,A}(k) = Pr[R \leftarrow \{0,1\}^{2k}; b \leftarrow A(R) : b = 0]$$

Theorem: If OWFs exist, then PRGs exist [HILL]

Towards a PRG from a OWF f : $\{0,1\}^k \to \{0,1\}^k$

- Attempt #1: Alg1

  Draw random $x \leftarrow \{0,1\}^k$, output $R = x \circ f(x)$ (Note: $\circ$ denotes concantenation)

- Does it work?

  No: to test if $R = x \circ y$ is the output of Alg1, check if $y = f(x)$

Towards a PRG from a OWF f : $\{0,1\}^k \to \{0,1\}^k$

- Attempt #2: Alg2

  Let B : $\{0,1\}^k \to \{0,1\}$ be a function such that, on input f(x), no ppt A can guess B(x) non-negligibly better than with probability ½. More formally: for all ppt A, $|p_{B,A}(k) - ½|$ is negligible, where $p_{B,A}(k) = \Pr[x \leftarrow \{0,1\}^k; b \leftarrow A(f(x)) : b = B(x)]$

  Such a function B is also known as a "hardcore bit," aka Goldreich-Levin bit. (Same Levin, originally from Dnipro, as in the Cook-Levin theorem that SAT in NP-complete.)

  Alg2: Draw random $x \leftarrow \{0,1\}^k$, output $R = f(x) \circ B(x)$

- Does it work?

  Not quite: (1) output of Alg2 is only k+1 bits long
  (2) f(x) does not necessarily look random!
  (3) does B even exist? YES: Goldreich-Levin theorem (see book)

Towards a PRG from a OWF f : $\{0,1\}^k \to \{0,1\}^k$

- Attempt #3: Alg3

  Let f be a one-way permutation. If x is uniformly random, then f(x) is as well
  Let B : $\{0,1\}^k \to \{0,1\}$ be a hardcore bit of f

  Draw random $x \leftarrow \{0,1\}^k$, output $R = f(x) \circ B(x)$

- Does it work?

  It's better than Alg2: (1) output is still only k+1 bits long
  But now (2) f(x) is random!

Towards a PRG from a OWF f : $\{0,1\}^k \to \{0,1\}^k$

- Attempt #4: Alg4 = Blum-Micali PRG

  Let f be a one-way permutation. If x is uniformly random, then f(x) is as well
  Let B : $\{0,1\}^k \to \{0,1\}$ be a hardcore bit of f

  Draw random $x \leftarrow \{0,1\}^k$, then:
  Let $x_0 = x$
  For i = 1 to 2k:
  $R_i = B(x_{i-1})$
  $x_i = f(x_{i-1})$
  output $R = R_1 \circ R_2 \ldots \circ R_{2k}$

- Does it work?

  Yes. Proof by "hybrid argument" – series of experiments in which some bits of R are computed as here, and other bits are truly random.

  (If we had another lecture we would use it to study the hybrid argument! Fundamental to cryptography research.)

Towards a PRG from a OWF f : $\{0,1\}^k \to \{0,1\}^k$

- Attempt #4: Alg4 = Blum-Micali PRG

  Let f be a one-way permutation. If x is uniformly random, then f(x) is as well
  Let B : $\{0,1\}^k \to \{0,1\}$ be a hardcore bit of f

  Draw random $x \leftarrow \{0,1\}^k$, then:
  Let $x_0 = x$
  For i = 1 to 2k:
  $R_i = B(x_{i-1})$
  $x_i = f(x_{i-1})$
  output $R = R_1 \circ R_2 \ldots \circ R_{2k}$

- BUT: Only works if f is a permutation – does not work for a general OWF.

  Why? Problem-solving session on Tue: show that f(f(x)) may be a constant function!

  .

  .

Towards a PRG from a OWF f : $\{0,1\}^k \to \{0,1\}^k$

- Attempt #5: HILL (very high-level explanation)

  Let f be a one-way permutation. If x is uniformly random, then f(x) is as well
  Let B : $\{0,1\}^k \to \{0,1\}$ be a hardcore bit of f

  High-level idea: HILL construct the function g such that the following is a PRG:

  Fix public parameters $r_1, \ldots, r_{2k}$
  Draw random $x \leftarrow \{0,1\}^k$, then:
  Let $x_0 = x$
  For i = 1 to 2k:
  $R_i = B(x_{i-1})$
  $x_i = f(g(x_{i-1}, r_i))$
  output $R = R_1 \circ R_2 \ldots \circ R_{2k}$

Why Is It Sufficient to Assume OWFs?
Чому достатньо припускати існування OWF?

- Can construct (symmetric) secure encryption from one-way functions.
  Ми знаємо, як побудувати (симетричне) безпечне шифрування з односторонніх функцій
- Roadmap / план:
  - Construct a pseudorandom generator from a one-way function
    будуємо псевдовипадковий генератор із OWF
  - Construct a pseudorandom function from a pseudorandom generator
    будуємо псевдовипадкову функцію з псевдовипадкового генератору
  - Construct a block cipher from a pseudorandom function
    будуємо блоковий шифр із псевдовипадкової функції
  - Construct secure symmetric encryption from a block cipher
    будуємо безпечне симетричне шифрування з блокового шифру

Pseudorandom Function: Definition
Псевдовипадкова функція: визначення

- Let $F : \{0,1\}^k \times \{0,1\}^k \rightarrow \{0,1\}^k$ be an efficiently computable function defined for every k

  Intuition: the output of a pseudorandom function looks like the output of a truly random function. Like “genie in a box” that, in response to a query x, selects a truly random string y.

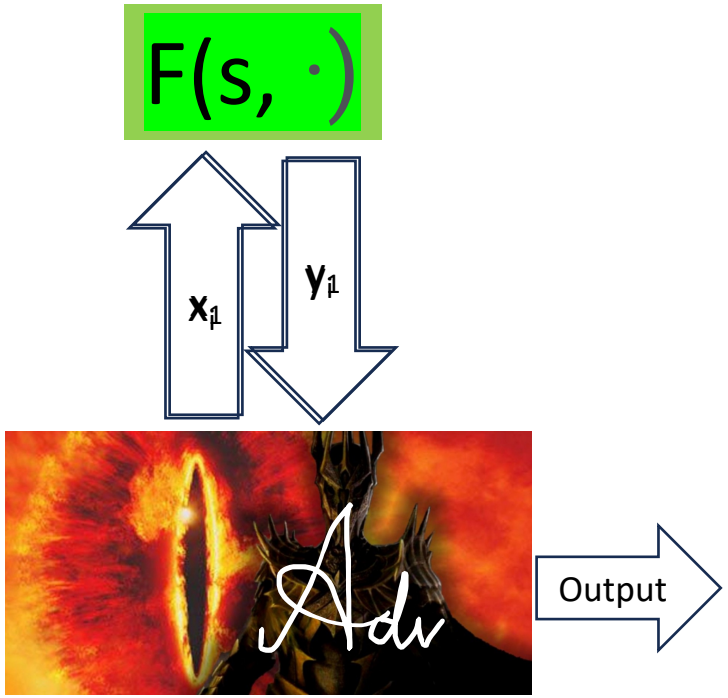
  Formally: F is a pseudorandom function if for every ppt A, $|p_{F,A}(k) - p_{Rand,A}(k)|$ is negligible, where $p_{F,A}(k)$ and $p_{Rand,A}(k)$ are the probability A outputs 0 experiments 1 and 2 respectively (in the next slide)

Pseudorandom Function: Definition
Псевдовипадкова функція: визначення

- Experiment 1: querying F

  pick seed s <- $\{0,1\}^k$

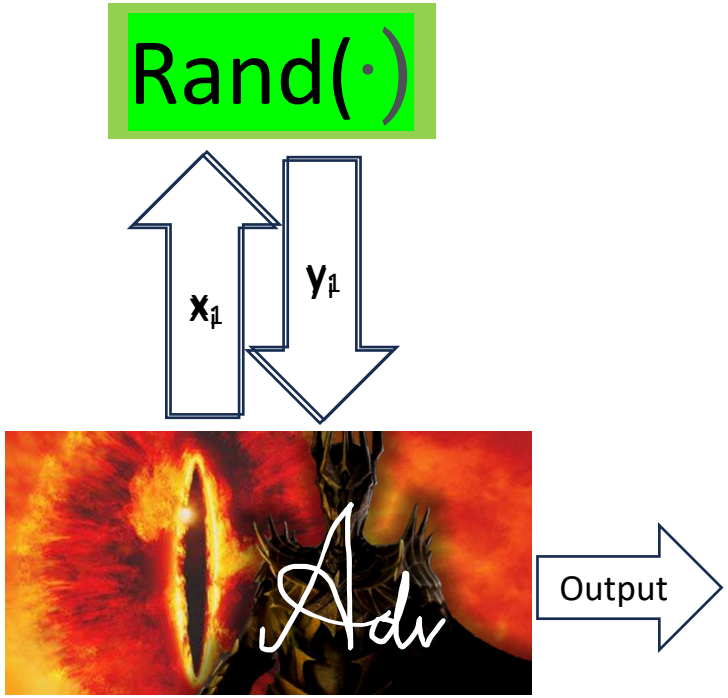
  interact with Adv:



- Experiment 2: querying Rand

pick a truly random function Rand

interact with Adv:



Theorem: Given a PRG, we can construct a pseudorandom function (PRF) [GGM = Goldreich, Goldwasser, Micali]

GGM Construction

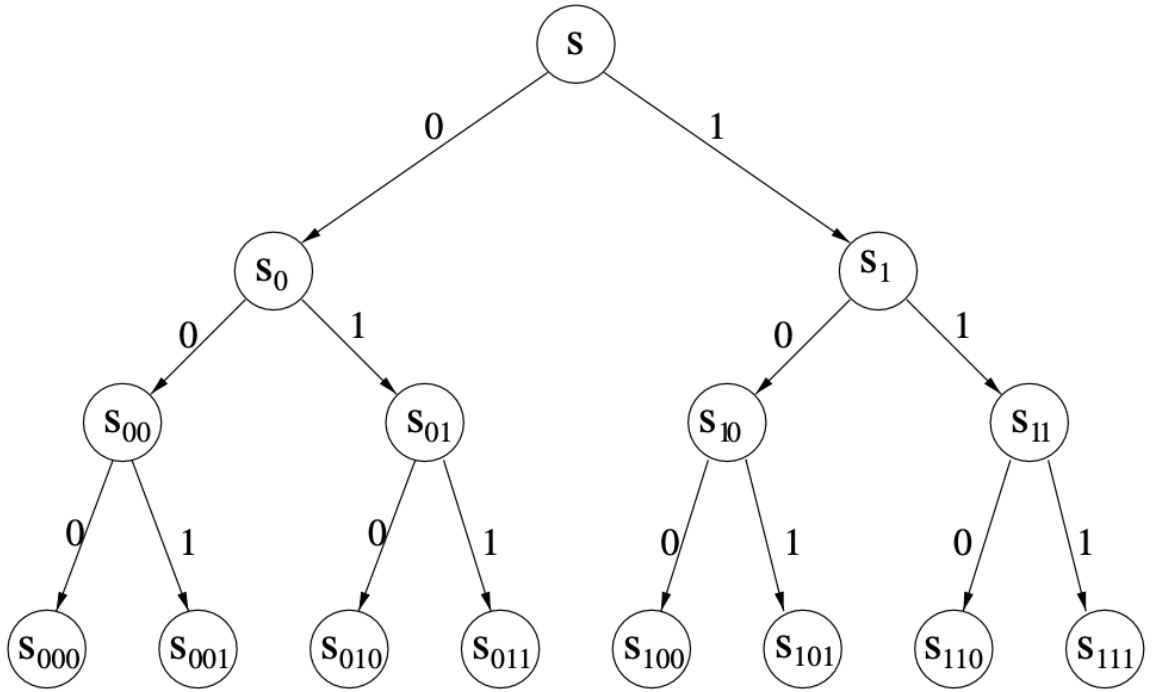
- Given a PRG $G : \{0,1\}^k \to \{0,1\}^{2k}$, let F be defined as follows:
  Let $G_0(s)$ denote the first k bits of G(s), $G_1(s)$ denote the next k bits

  For a bit string x, integer i, let bit(x,i) be the i^{th} bit of x
  prefix(x,i) be the i-bit prefix of x
  prefix(x,0) = ε (the empty string)
  prefix(x,k) = x
  LSB(x) be the least significant bit of x

On input seed s, query x:
Let s_ε = s (recall that ε denotes the empty string)
for i from 1 to k,
$s_{prefix(x,i)} = G_{bit(x,i)}(s_{prefix(x,i-1)})$
Output s_x

GGM Construction

We let $s_\lambda = s$ and $s_{\alpha\sigma} = G_\sigma(s_\alpha)$. The value of $f_s(\sigma_1\sigma_2\cdots\sigma_n) = s_{\sigma_1\sigma_2\cdots\sigma_n}$ is obtained at the leaf reachable from the root (labeled s) by following the path $\sigma_1\sigma_2\cdots\sigma_n$.



For example, $f_s(001) = s_{001} = G_1(s_{00}) = G_1(G_0(s_0)) = G_1(G_0(G_0(s)))$.

Figure 3.5: Construction 3.6.5, for $n = 3$

GGM Construction – Proof of Security

- On input seed s, query x:
  Let s_ε = s (recall that ε denotes the empty string)
  for i from 1 to k,
  $s_{prefix(x,i)} = G_{bit(x,i)}(s_{prefix(x,i-1)})$
  Output s_x

- Wish to show: If Adv can distinguish Experiment 1 with the GGM PRF from Experiment 2, then G is not a PRG
- Proof idea: two-dimensional hybrid argument – change the experiment so that
  dimension 1: the seeds at increasing depth are truly random
  dimension 2: the first set of queries are answered using random seeds that sit deeper.
- (This is cool, but we don't have time to see it ☹)

Why Is It Sufficient to Assume OWFs?
Чому достатньо припускати існування OWF?

- Can construct (symmetric) secure encryption from one-way functions.
  Ми знаємо, як побудувати (симетричне) безпечне шифрування з односторонніх функцій
- Roadmap / план:
  - Construct a pseudorandom generator from a one-way function
    будуємо псевдовипадковий генератор із OWF
  - Construct a pseudorandom function from a pseudorandom generator
    будуємо псевдовипадкову функцію з псевдовипадкового генератору
  - Construct a block cipher from a pseudorandom function
    будуємо блоковий шифр із псевдовипадкової функції
  - Construct secure symmetric encryption from a block cipher
    будуємо безпечне симетричне шифрування з блокового шифру

Block cipher, aka as pseudorandom permutation: Definition

Блоковий шифр або псевдовипадкова перестановка: визначення

- Let $P : \{0,1\}^k \times \{0,1\}^k \rightarrow \{0,1\}^k$ be an efficiently computable function defined for every k, such that $P^{-1}(K,\cdot)$ is also efficiently computable

  Intuition: just like a pseudorandom function, but it's a permutation, and you can query it in both directions. Indistinguishable from a truly random permutation that's a "genie in a box."

  Notation: E_K denotes $P^{-1}(K,\cdot)$, while D_K denotes $P^{-1}(K,\cdot)$

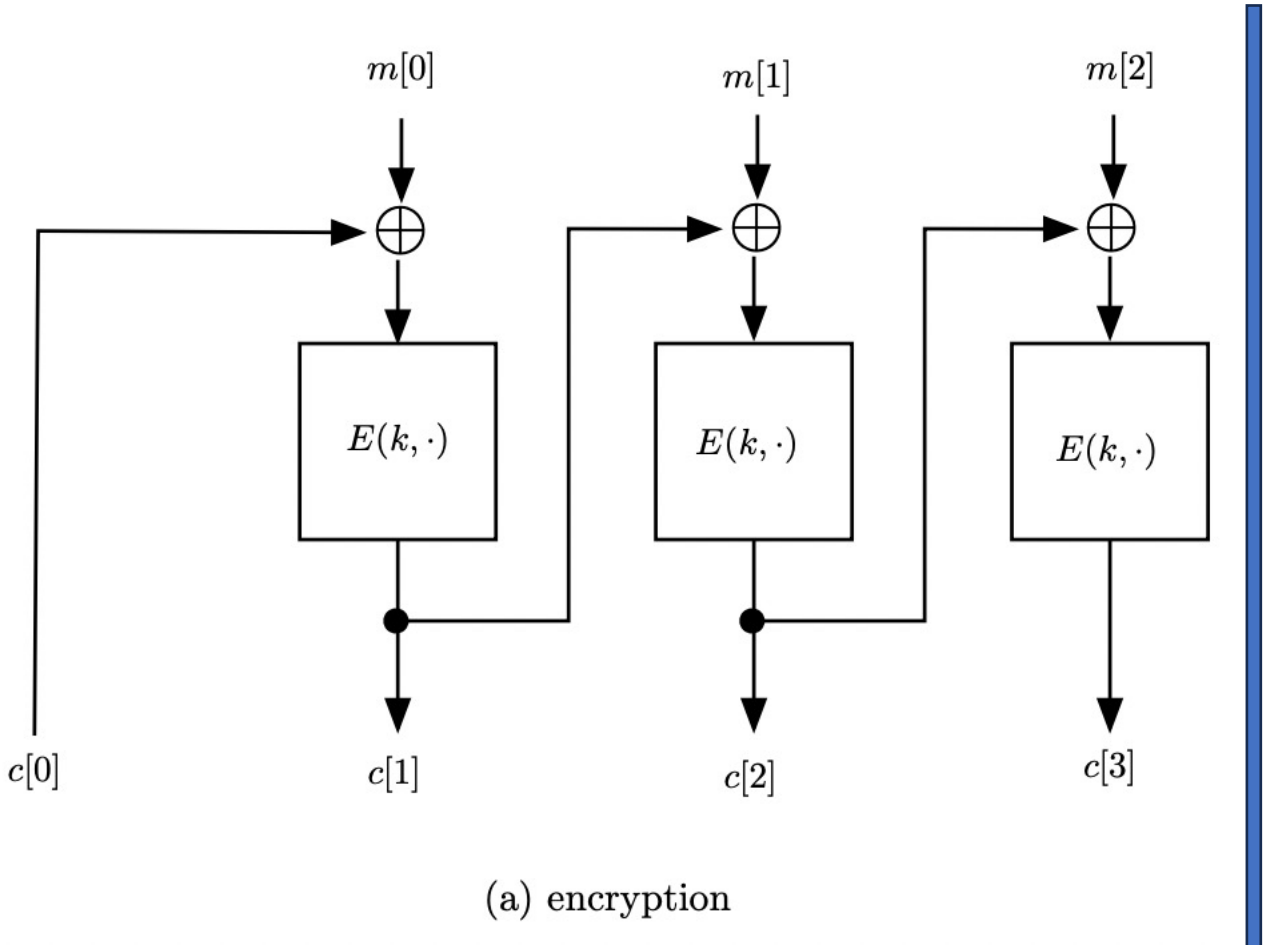
- See book's Section 3.7 for formal definition and construction

Why Is It Sufficient to Assume OWFs?
Чому достатньо припускати існування OWF?

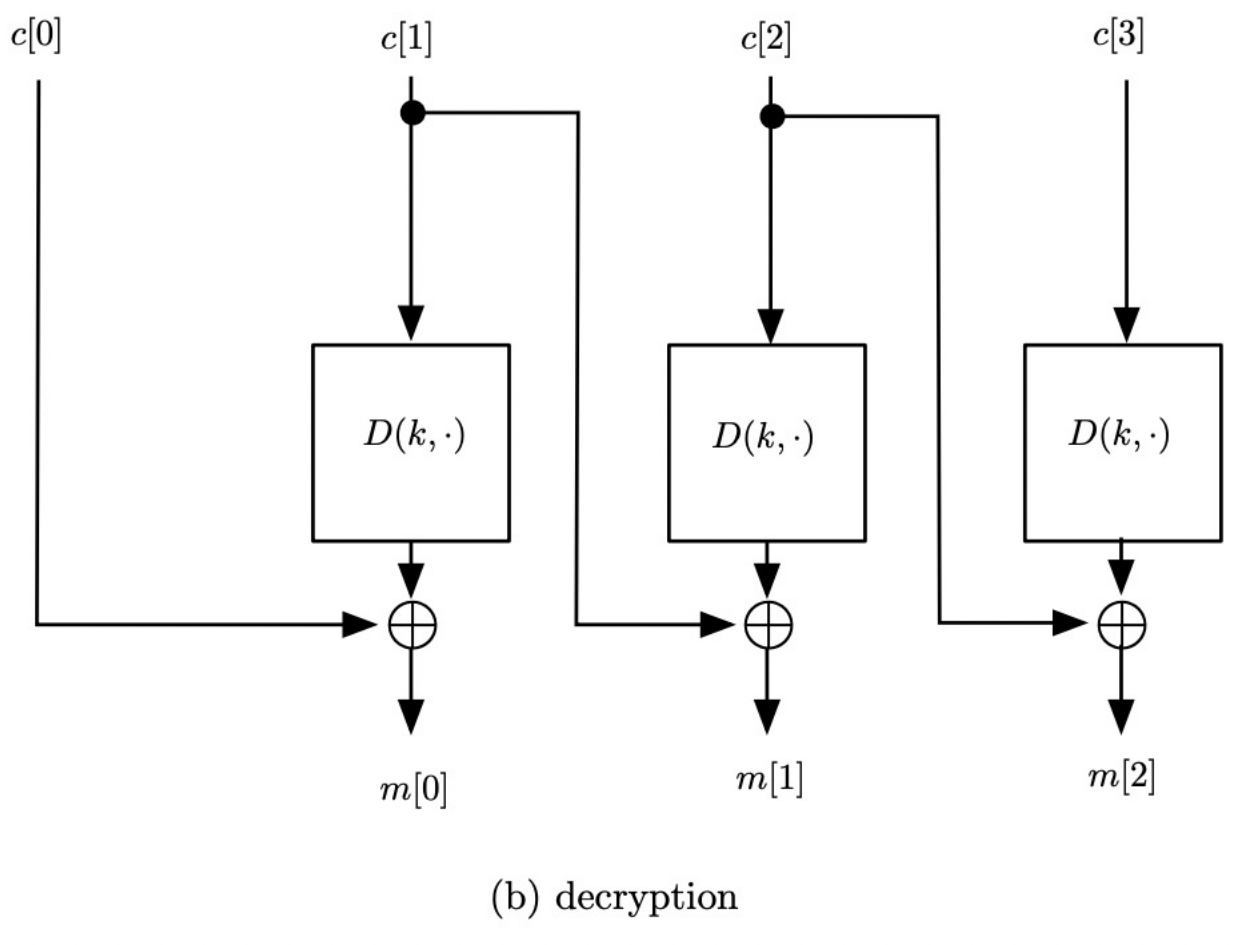
- Can construct (symmetric) secure encryption from one-way functions.
  Ми знаємо, як побудувати (симетричне) безпечне шифрування з односторонніх функцій
- Roadmap / план:
  - Construct a pseudorandom generator from a one-way function
    будуємо псевдовипадковий генератор із OWF
  - Construct a pseudorandom function from a pseudorandom generator
    будуємо псевдовипадкову функцію з псевдовипадкового генератору
  - Construct a block cipher from a pseudorandom function
    будуємо блоковий шифр із псевдовипадкової функції
  - Construct secure symmetric encryption from a block cipher
    будуємо безпечне симетричне шифрування з блокового шифру

How to encrypt using a block cipher
Як зашифрувати за допомогою блокового

- CBC mode (no longer used, some issues…) – HW problem is to break & fix it



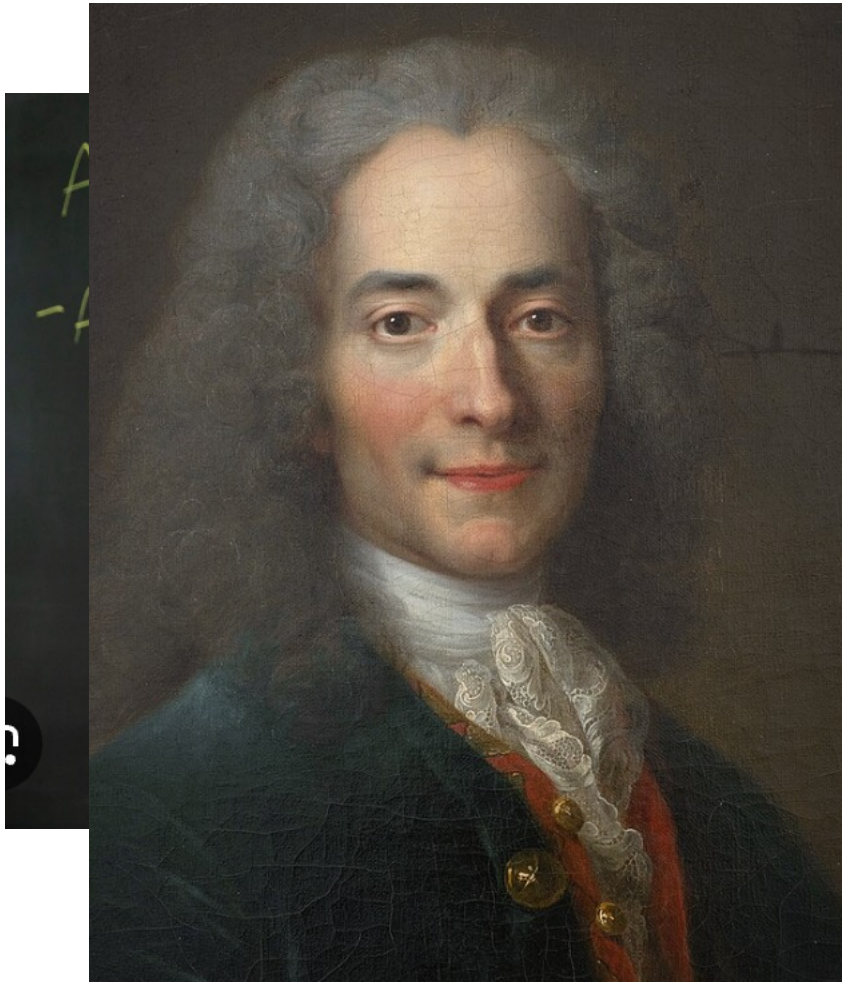
(a) encryption



(b) decryption

Public-Key Encryption From OWFs?

- No! Impagliazzo's worlds:
  - Algorithmica: P=NP. Heuristica: almost the same
  - Pessiland: P does not equal NP, but OWFs don't exist
  - Minicrypt: OWFs exist, but PK encryption does not
  - Cryptomania: PK encryption and beyond
- Cryptomania – next lecture



Digital Signatures From OWFs?

- Yes. We will also talk about this in our next lecture.

Reading

- Use Goldreich's book "Foundations of Cryptography, Part 1" for general reference
- Read 2.4.2, 2.4.3, and 2.2.4 for Wednesday.

Homework: work on it with Illia tomorrow, hand it in before Wednesday's class

- (1) Prove that if P=NP, then encryption is impossible (slide 26)
- (2) Prove that Encrypt(K,1^λ,m; random bits R) is a one-way function of K,R. (Hint: m can be a fixed message, known to the adversary; for example, you can use the all-0 string for m.)
- (3) Suppose f is a OWF. Design a OWF f' such that f'(f'(x)) = 0^k for all x
- (4) Suppose G_1 and G_2 are PRGs. Is G'(s) = G_1(s) XOR G_2(s) a PRG?
- (5) Suppose F(s,x) is a PRF. Is F(x,s) a PRF?
- (6) Why doesn't the CBC mode encryption scheme satisfy the definition of security? How do you construct an encryption scheme that does?